Eur päisches Patentamt	Eur pean Patent Office	Office européen des brevets

Bescheinigung	Certificate	Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr.	Patent application No.	Demande de brevet n°

02021035.7

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

[illegible]

R C van Dijk

Anmeldung Nr:
Application no.: 02021035.7
Demande no:

Anmeldetag:
Date of filing: 20.09.02
Date de dépôt:

Anmelder/Applicant(s)/Demandeur(s):

SIEMENS AKTIENGESELLSCHAFT
Wittelsbacherplatz 2
80333 München
ALLEMAGNE

Bezeichnung der Erfindung/Title of the invention/Titre de l'invention:
(Falls die Bezeichnung der Erfindung nicht angegeben ist, siehe Beschreibung.
If no title is shown please refer to the description.
Si aucun titre n'est indiqué se referer à la description.)

Method for improving the performance of 3-dimensional concatenated product codes

In Anspruch genommene Prioriät(en) / Priority(ies) claimed /Priorité(s) revendiquée(s)
Staat/Tag/Aktenzeichen/State/Date/File no./Pays/Date/Numéro de dépôt:

Internationale Patentklassifikation/International Patent Classification/
Classification internationale des brevets:

H03M13/29

Am Anmeldetag benannte Vertragstaaten/Contracting states designated at date of filing/Etats contractants désignées lors du dépôt:

AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LU MC NL PT SE SK TR

EPO - Munich
3
20. Sep. 2002

Description

Method for improving the performance of 3-dimensional concatenated product codes

The present invention relates to a method for of 3-dimensional concatenated product codes.

The capacity of optical transmission systems has rapidly increased in the last several years. The ability to upgrade a low bitrate system to a higher bitrate system by improving the optical components and compensating the limiting physical effects was the key for achieving system evolution.

The introduction of error control coding (ECC, FEC) was a very efficient tool that successfully improved the performance and reliability of digital data transmission. Adding redundancy check bits to the information bits and advanced decoding provides the possibility of increasing the transmission distance and further makes the system more robust to adverse conditions impairing transmission performance such as temperature variations and acoustic vibrations. Because of complexity reasons hard decoding is preferred to soft decoding. Codes like Bose-Chaudhuri-Hocquenghem codes (BCH) or extended BCH codes, which can be implemented easily, are preferred. To improve coding gain concatenated codes are used. But this group of codes is very sensitive when special error patterns are received. Those error patterns cannot be corrected and thus lead to an increase of the bit error rate (error flaring).

The basics of product coding are explained in "Prüfbare und korrigierbare Codes" W. Wesley Peterson, Oldenburg Verlag 1967, Seiten 117- 123.

Also 3-dimensional product codes can be used for further improvement. Each bit and therefore each error participates in three equations .

This invention refers to a method for improving the performance of 3-dimensional concatenated product codes and for the reduction of the error flaring.

The present invention provides a method for improving the bit error rate (BER) and therefore the coding gain according to claim 1. This is achieved by applying an encoding procedure, then interleaving the information bits and (at least a group of) check bits and finally applying an inner encoding procedure, whereby at least one of the codes is a with 3-dimensional product code. The best performance is gained with 3-dimensional outer product code, an interleaver according to this invention and an inner 3-dimensional outer product code.

The interleaver should be realizable with low design complexity and memory requirement.

Applying the described interleaving procedure both on columns and rows and regarding the first layer of a 3-dimensional code, it breaks the error bursts in rows and in columns of this layer.

If the interleaving is different in each of the layers having the same orientation even the "error burst" in the third dimension is broken.

Another advantageously interleaving method is the shifting of the parallel layers of the code matrix by different bits and than shifting the rows or columns by different numbers.

In order to achieve an easy implementable interleaver, for example the first column of a first layer of the 3-dimensional code remains unchanged and the elements (bits) of the following columns are cyclically shifted by one, two, three etc. positions, so that the elements of the first row are translated into diagonal elements by the interleaver. Then, after this first interleaving step, the positions of the elements of the first row remain unchanged and the elements of the following rows are shifted by one, two, three etc. positions.

For the following layer the position of the elements of the first row are shifted by one position and the elements of the following rows by 2, 3, 4 etc. positions (columns, rows and layers can be interchanged).

Another interleaving procedure start shifting the layers by 1, 2, 3, etc. positions and proceeds with shifting of rows or columns rectangular oriented to these layers by 0, 1, 2, 3 etc. positions.

For the three dimensional code a BCH-code with one or two error correction possibility is preferred.

The invention will became more apparent with reference to the following description along with the accompanying drawings.

Figure 1 schematic of a concatenated coding system;
Figure 2 schematic of a three-dimensional product code;
Figure 3 permanent error-pattern;
Figure 4 exemplary elements of a code matrix;
Figure 5 an example of a shift procedure;
Figure 6 exemplary elements of a code matrix and an interleaved code matrix according to this example;
Figure 7 exemplary elements of a code matrix and an interleaved code matrix;
Figure 8 a second example for an interleaved code matrix.

Figure 1 shows a schematic of a transmission system with a concatenated code implementation. The information bits "a" are fed to the input 1 of an outer encoder 2, which is the first element of a serial concatenation including an interleaver 3 and an inner encoder 4. At least one of the codes is supposed to be 3-dimensional product code, whereas the other code can be a 1- or 2-dimensional product code. But for an optimal success, the outer and the inner code should be 3-dimensional product code.

The information bits "a" and the generated check bits "c" are fed to a modulator 5, which converts the bits into physical signals "s" being transmitted over the transmission path 6 to a demodulator 7 at the receiving side. Because of the non ideal transmission path the signals are disturbed by signal impairments SI, e.g. external perturbations or physical effects of the transmission path. The demodulator 7 converts the received signals into (binary) bits "r", which are fed to a serial concatenation of an inner decoder 8, a deinterleaver (inverse interleaver) 9 and an outer decoder 10. The corrected information bits a_{COR} are emitted at the output 11.

In **figure 2** a code matrix of a three-dimensional product code is shown, which may be generated by the outer encoder 2. The code matrix has the dimension of N × N × N bits and contains K × K × K information bits, the generated check bits C_R, C_C and C_T allow the correction of at least one error for each code vector (columns or row).

The information bits and the check bits form code vectors $V_{i,j}$; $V_{i,k}$; $V_{j,k}$, each code vector $V_{i,j}$; $V_{i,k}$; $V_{j,k}$ containing a string of the adjoining information bits ($a_{i,j,f(k)}$; $a_{i,f(j),k}$; $a_{f(i),j,k}$) and adjoining check bits $C_T = c_{i,j,f(k)}$; $C_C = c_{i,f(j),k}$; $C_R = c_{f(i),j,k}$. For example, the code vector $V_{j,k}$ contains the information bits $a_{f(i),j,k}$, where j, k = constant and for all i = 1 - K bits, and the checkbits $C_R = c_{f(i),j,k}$ for i = (K+1) - N and j, k = constant.

Checks on check-bits CC can be used for checking the check-bits. Of course, also a non square code word matrix and also different codes can be used for rows and columns. The indices of the code elements, information and check bits, are consecutively numbered for each dimension.

In **figure 3** an example of a permanent 8-error-event for a product code with one-error-correcting codes as component codes is shown. In case of two errors in one row or one column a one-error-correcting code is overloaded and its decoder will, with high probability, add new errors. In the shown 8 error event also the product code will be overloaded because two errors occur in all relevant code vectors. Hence the errors will never be corrected by the product code alone, no matter how many iterations are used. The error pattern is permanent and leads to error flaring. For component codes that can correct two or more errors corresponding permanent error patterns exist.

However, such error patterns, which are permanent with regard to the product code alone, may be resolved in the overall concatenation thanks to the interleaver and the inner coding and decoding stages.

We consider the three dimensional product code word in **figure 4**. The code elements a, c are replaced by numbers, representing their original bit sequence. The front of the cube shows a first X-layer X1 (index k = 1, constant). The following layers (parallel slices) are numbered X2 - X5. The layers Y contains all code elements with an constant i, where i is 1 - 5, and the layer Z contains all code elements with an constant j, where j is 1 - 5.

Figure 5 shows one of the possible interleaving methods. The possible shift operations of code elements are described by letters X, Y, Z according to the layers and the directions for the shift of code elements or layers are indicated by ciphers 1 - 4. Regarding this first example for an interleaving process, the Y-layers (i-constant-layers) containing code elements with a constant index i for each layer are shifted by 0, 1 ,2 3 and 4 positions. Than the code elements shown in under the original code matrix are inserted in the upper part of the cube. In this example only the elements of all the i-rows (respective Z-layers) are interleaved while the columns in the j-direction direction still contain the same bits and the rows in the k-direction are unchanged.

Figur 6 shows the elementary code elements before and after this shift operation in the form of a table. Each X-layer (k = 1, 2, 3, 4, 5) contains 25 numbers. Only the elements of the columns of all X-layers are shifted by the same numbers, respectively all Y-layers are shifted by 0, 1, 2, 3, 4 positions in the Y2 direction. So all rows of the X-layers and all (horizontal) rows of the Z-layers (respectively Y-layers) still contain the same bits. This interleaving is not very efficient. This interleaving procedure is helpful against burst errors but ineffective against the error pattern shown in figure 3.

An efficient interleaving procedure would shift the code elements different in each layer and add an additional shift procedure to interleave the code elements of the columns.

An efficient procedure is shown in table **figur 7**. In the first X1-layer (k = 1) the positions of the elements in the columns 1 - 5 are shifted by 0, 1, 2, 3 and 4 positions. Than the positions in the rows 1 - 5 are shifted by 0, 1, 2, 3 and 4 positions.

In the next X2-layer (k = 2) the positions of the elements in the columns 1 - 5 are again shifted by 0, 1, 2, 3 and 4 positions but than the elements in all rows are shifted by 1, 2, 3, 4, 0 positions etc.

This corresponds with first shifting all Y-layers are by 0, 1, 2, 3, 4 positions according to fig. 6, than shifting the rows of the X-layers by different values and different for each X-layer.

After the complete interleaving every code vector (rows and colums) contains only one code element of the first code matrix.

Another interleaving possibility is shown in **figure 8**. In a first interleaving step all the Y layers are shifted in the Y1 direction (figure 5) by 0 - 4 positions and than the i-rows rectangular to this Y-layers are shifted by 0 to 4 positions for the first new X-layer X1, 1 to 0 positions for the second X-layer X2 (modulo N, according to the number of shifted code elements), 2 to 1 positions for the third X-layer etc. After the complete interleaving every code vector again contains only one code element of the first code matrix. This interleaving is a good solution for burst errors because the adjacent bits of the A, C cube are separated very well.

Of course, the interleaving could start with every kind of layers to reach similar results. The sequence in with the code elements are transmitted must be taken into account for burst correction abilities. Also the sequence of shifting layers, or code elements by 0 - 4 positions (in the example) can by changed to a random sequence, but the shift procedure must be different for each layer, row or column.

Claims

1. Method for improving error correction of concatenated codes comprising the steps of
storing information bits (a_{ijk}), which form a cuboid information matrix ($A = a_{ijk}$; i, j, k = 1, 2, …n),
generating check bits (c_{ijk}) of said cuboid information matrix (A) by an outer code to obtain first code vectors ($V_{i,j}$; $V_{i,k}$; $V_{j,k}$), each first code vector ($V_{i,j}$; $V_{i,k}$; $V_{j,k}$) containing a string of the adjoining information bits ($a_{i,j,f(k)}$; $a_{i,f(j),k}$; $a_{f(i),j,k}$) and said check bits ($C_T = c_{i,j,f(k)}$; $C_C = c_{i,f(j),k}$; $C_R = c_{f(i),j,k}$),
the information matrix (A) and the check bits forming a code matrix (A, C),
cyclically interleaving the information bits (a_{ijk}) and respectively the check bits (c_{ijk}) to obtain an interleaved code matrix ($B, C^* = b_{ijk}, c^*_{ijk}$) with second code vectors (W_{ij}; W_{ik}; W_{jk}), whereby the second code vectors (W_{ij}; W_{ik}; W_{jk}) of the interleaved code matrix ($B, C^* = b_{ijk}, c^*_{ijk}$) contain only one information bit (a_{ijk}) of each corresponding first code vector ($V_{i,j}$; $V_{i,k}$; $V_{j,k}$) and
coding the bits of the interleaved code matrix (B, C^*) by an inner code,
where at least the outer code or the inner code is a three dimensional product code.

2. Method according to claim 1,
wherein the outer and the inner code are three dimensional product codes

3. Method according to claim 1 or 2,
wherein said stored information bits (a_{ijk}) (i, j , k = 1,.2, ...5) form a cube information matrix (A).

4 Method according to claim 1, 2 or 3,
wherein the interleaving of the information bits (a_{ijk}) and respectively the check bits (c_{ijk}) comprises the steps of

cyclically shifting said information bits in columns (j = 1, 2, ...) and in rows (i = 1,2, ...) by different values (0, 1, 2, ..., n) and different for each parallel layer (X1 - X5-layer: k= const. 1, 2, 3, 4, 5) having the same orientation of said interleaved code matrix (A, C) to obtain said interleaved code matrix (B, $C^* = b_{ijk}, c_{ijk}$), whereby each second code vector (W_{ij}; W_{ik}; W_{jk}) of the interleaved code matrix (B, $C^* = b_{ijk}, c_{ijk}$) contains only one information bit (a_{ijk}) of each corresponding first code vector ($V_{i,j}$; $V_{i,k}$; $V_{j,k}$).

5. Method according to claim 1, 2 or 3,
wherein the interleaving of the information bits (a_{ijk}) and respectively the check bits (c_{ijk}) comprises the steps of cyclically shifting said information bits (a_{ijk}) and respectively the check bits (c_{ijk}) for each parallel layer (j = const.1, 2, 3, 4, 5 or k = const.1, 2, 3, 4, 5) of said of said code matrix (A, $C = a_{ijk}, c_{ijk}$) having the same dimensions by different values (0, 1, 2, ..., n) to obtain a first code matrix and than shifting the rows or respectively columns of rectangular layers by different values and different for each rectangular layer - or vice versa - to obtain an interleaved code matrix (B, $C^* = b_{ijk}, c_{ijk}$), whereby each second code vector (W_{ij}; W_{ik}; W_{jk}) of the interleaved code matrix (B, $C = b_{ijk}, c_{ijk}$) contains only one information bit (a_{ijk}) of each corresponding first code vector ($V_{i,j}$; $V_{i,k}$; $V_{j,k}$)

6. Method according to claim 1 to 5,
wherein the number of shift positions is altered by 1 from a row to the next row - a column to the next column - a layer to the next layer.

7. Method according to one of the claims,
comprising the steps of decoding the interleaved code matrix by an inner code,
deinterleaving the code matrix and decoding the code matrix by an inner code.

8. Method according to claim 7,
using an iterative decoding procedure.

200210992

EPO - Munich
3
20. Sep. 2002

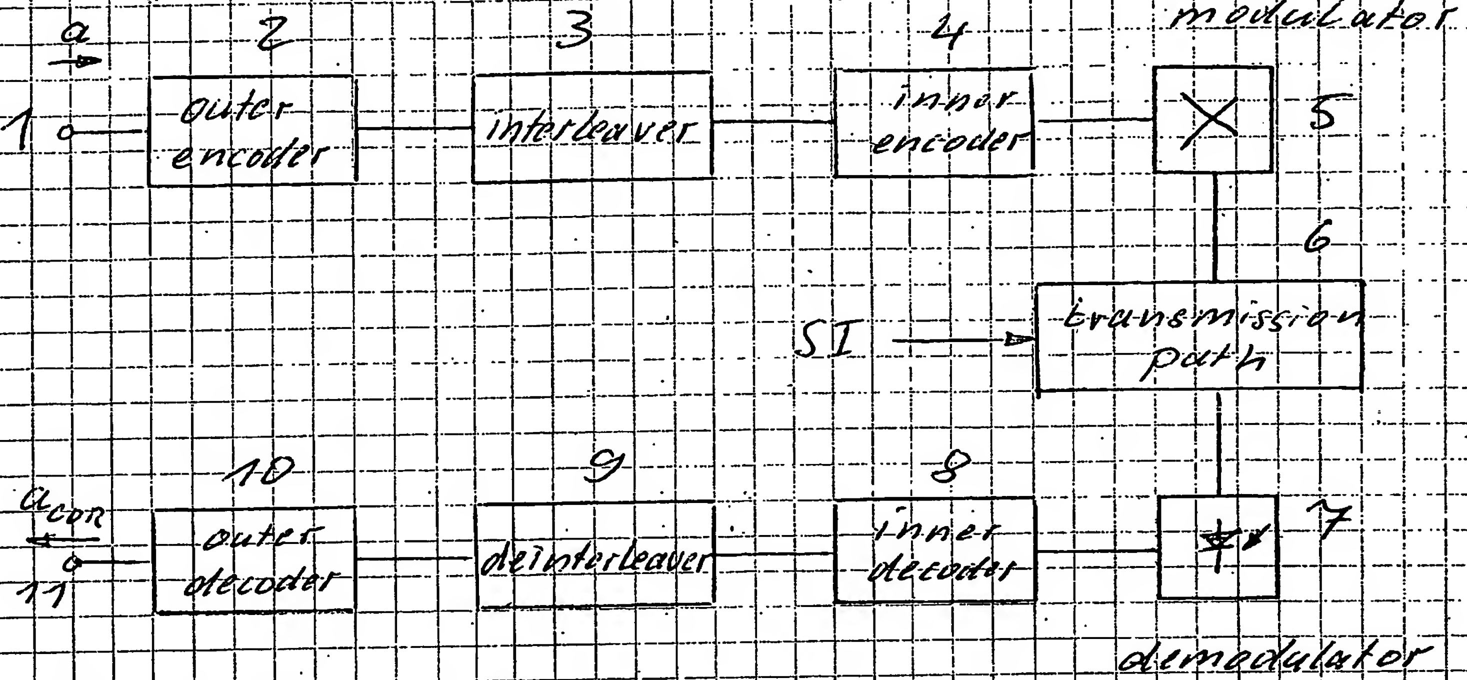
Abstract

Method for improving the performance of concatenated codes

The method implies the steps of generating check bits (c) from information bits (a), which are represented by an information matrix, by an outer code,
shifting cyclically the information bits (a) and the check bits (c) to obtain an interleaved code matrix, and
than coding the bits of the interleaved code matrix by an inner code, where at least the outer code or the inner code is a product code.

Fig. 1

Fig. 1


a
1
2
outer encoder
3
interleaver
4
inner encoder
modulator
5
6
SI
transmission path
10
9
8
7
a_{corr}
11
outer decoder
deinterleaver
inner decoder
demodulator

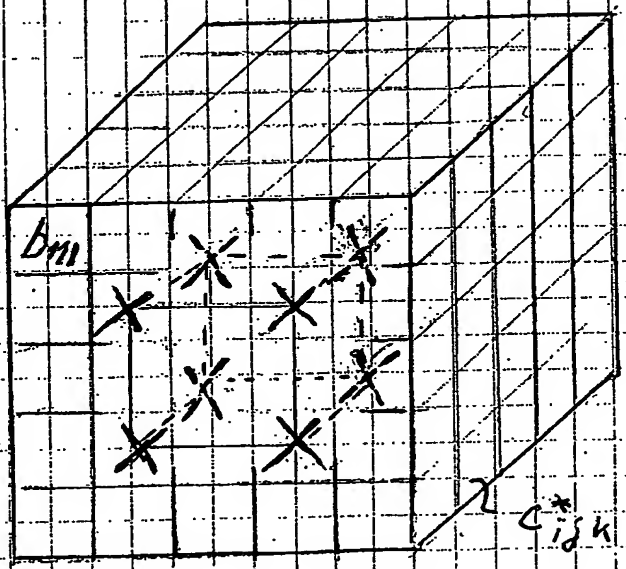

b_{m1}
c^*_{ijk}

Fig 3

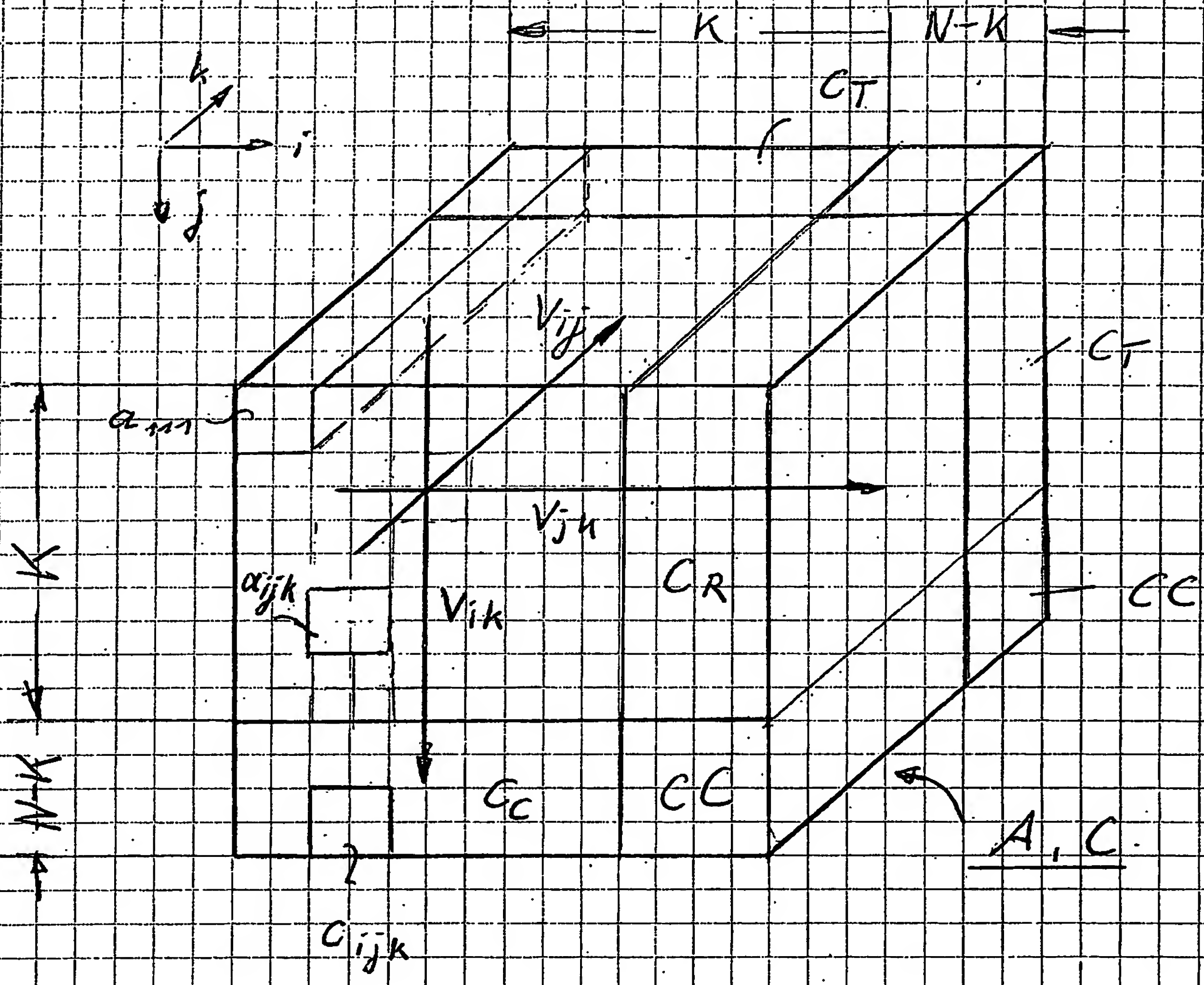
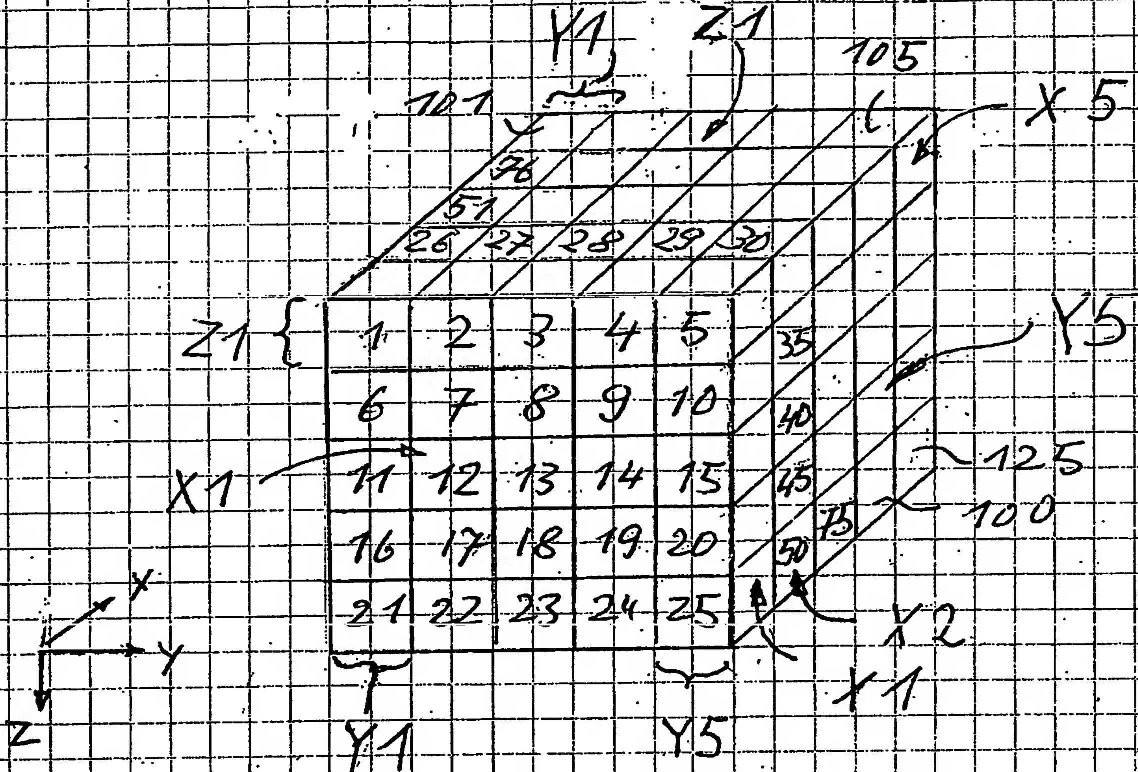


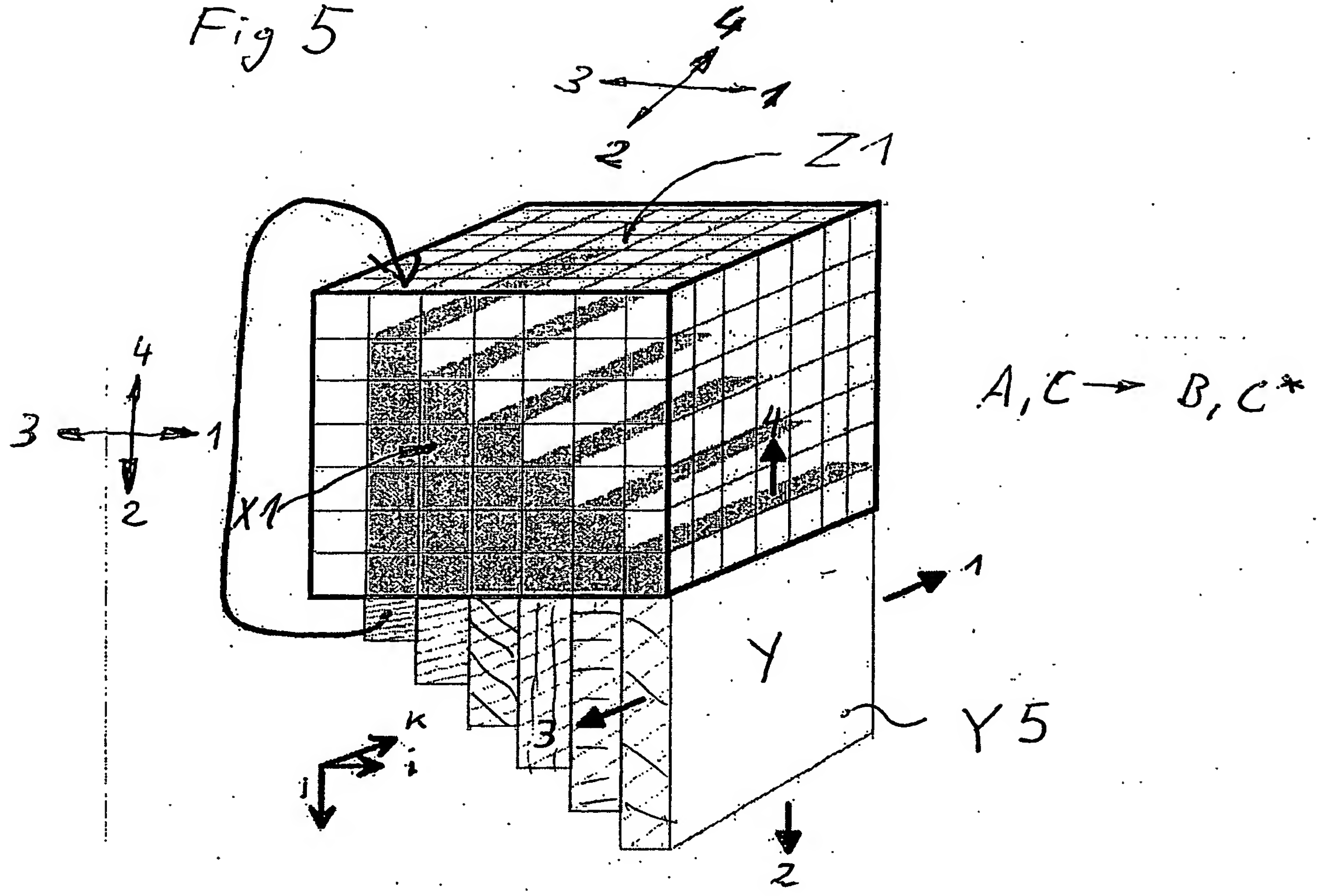
Fig 2



Y1
Z1
105
101
X5
76
51
26
27
28
29
30
Z1
1
2
3
4
5
35
6
7
8
9
10
40
Y5
X1
11
12
13
14
15
45
125
16
17
18
19
20
50
75
100
21
22
23
24
25
X2
X1
X
Y
Z
Y1
Y5

Fig 4

Fig 5



A, C					B, C*				
		$k=1$							
1	2	3	4	5	1	22	18	14	10
6	7	8	9	10	6	2	23	19	15
11	12	13	14	15	11	7	3	24	20
16	17	18	19	20	16	12	8	4	25
21	22	23	24	25	21	17	13	9	5
		$k=2$							
26	27	28	29	30	26	47	43	39	35
31	32	33	34	35	31	27	48	44	40
36	37	38	39	40	36	32	28	49	45
41	42	43	44	45	41	37	33	29	50
46	47	48	49	50	46	42	38	34	30
		$k=3$							
51	52	53	54	55	51	72	68	64	60
56	57	58	59	60	56	52	73	69	65
61	62	63	64	65	61	57	53	74	70
66	67	68	69	70	66	62	58	54	75
71	72	73	74	75	71	67	63	59	55
		$k=4$							
76	77	78	79	80	76	97	93	89	85
81	82	83	84	85	81	77	98	94	90
86	87	88	89	90	86	82	78	99	95
91	92	93	94	95	91	87	83	79	100
96	97	98	99	100	96	92	88	84	80
		$k=5$							
101	102	103	104	105	101	122	118	114	110
106	107	108	109	110	106	102	123	119	115
111	112	113	114	115	111	107	103	124	120
116	117	118	119	120	116	112	108	104	125
121	122	123	124	125	121	117	113	109	105

Fig 6

Fig 7

A,C					B,C*				
		k=1							
1	2	3	4	5	1	22	18	14	10
6	7	8	9	10	15	6	2	23	19
11	12	13	14	15	24	20	11	7	3
16	17	18	19	20	8	4	25	16	12
21	22	23	24	25	17	13	9	5	21
		k=2							
26	27	28	29	30	35	26	47	43	39
31	32	33	34	35	44	40	31	27	48
36	37	38	39	40	28	49	45	36	32
41	42	43	44	45	37	33	29	50	41
46	47	48	49	50	46	42	38	34	30
		k=3							
51	52	53	54	55	64	60	51	72	68
56	57	58	59	60	73	69	65	56	52
61	62	63	64	65	57	53	74	70	61
66	67	68	69	70	66	62	58	54	75
71	72	73	74	75	55	71	67	63	59
		k=4							
76	77	78	79	80	93	89	85	76	97
81	82	83	84	85	77	98	94	90	81
86	87	88	89	90	86	82	78	99	95
91	92	93	94	95	100	91	87	83	79
96	97	98	99	100	84	80	96	92	88
		k=5							
101	102	103	104	105	122	118	114	110	101
106	107	108	109	110	106	102	123	119	115
111	112	113	114	115	120	111	107	103	124
116	117	118	119	120	104	125	116	112	108
121	122	123	124	125	113	109	105	121	117

A,C					B,C*				
		k=1							
1	2	3	4	5	1	102	78	54	30
6	7	8	9	10	35	6	107	83	59
11	12	13	14	15	64	40	11	112	88
16	17	18	19	20	93	69	45	16	117
21	22	23	24	25	122	98	74	50	21
		k=2							
26	27	28	29	30	55	26	2	103	79
31	32	33	34	35	84	60	31	7	108
36	37	38	39	40	113	89	65	36	12
41	42	43	44	45	17	118	94	70	41
46	47	48	49	50	46	22	123	99	75
		k=3							
51	52	53	54	55	104	80	51	27	3
56	57	58	59	60	8	109	85	56	32
61	62	63	64	65	37	13	114	90	61
66	67	68	69	70	66	42	18	119	95
71	72	73	74	75	100	71	47	23	124
		k=4							
76	77	78	79	80	28	4	105	76	52
81	82	83	84	85	57	33	9	110	81
86	87	88	89	90	86	62	38	14	115
91	92	93	94	95	120	91	67	43	19
96	97	98	99	100	24	125	96	72	48
		k=5							
101	102	103	104	105	77	53	29	5	101
106	107	108	109	110	106	82	58	34	10
111	112	113	114	115	15	111	87	63	39
116	117	118	119	120	44	20	116	92	68
121	122	123	124	125	73	49	25	121	97

Fig 8